Preface

The transmission links between devices in a computer network may introduce significant errors because the links are analog, whereas computing devices are digital. Moreover, the design error rates for transmission links are often much higher than those tolerated for digital computing devices. Computer hardware and software have been designed to provide recovery when errors occur on communication lines. In the first paper in this issue, Bryant takes this process one step further by continuously monitoring the telephone lines used for data transmission. Line quality monitoring improves network availability by locating line problems before they become critical and by recording the causes of intermittent errors. Experiments at one installation show that use of this method greatly increased circuit availability.

The two remaining papers in this issue discuss effective programming. In recent years, a number of new techniques have been developed to reduce the effort involved in designing, coding, documenting, and maintaining computer programs. These techniques, which are sometimes referred to as improved programming technologies, are frequently used in combination. The papers by Stay and Van Leer build on some of these technologies and represent attempts to refine the program design process even further.

Stay describes Hierarchical Function Design, a single method and format that allows an application system to be specified from the highest functional level of a conceptual design to the lowest detailed level in a coded routine. Hierarchical Function Design combines two recently developed technologies: structured design, which incorporates techniques for converting a problem description into a modular program structure, and Hierarchy plus Input-Process-Output (HIPO), a method of top-down systems design that was developed originally as a program documentation tool.

Van Leer is also concerned with program design. He approaches the problem at a more detailed level, using a tool called the Program Design Language (PDL). PDL was originally designed to teach structured programming, but experience showed that it provided a method for improving program design. Used with other modern programming technologies, such as structured walk-throughs, top-down implementation, and structured programming, PDL has contributed to increased programmer productivity.

George C. McQuilken
Editor